3Q－1

ＳＱＬの実行順序チェック方式 (DBprobe-check)

夏目 義久　加納 直哉　赤間 浩樹
ＮＴＴ情報通信研究所

1．はじめに

DBシステムの性能トラブルへの対策として、我々は実測に基づく性能評価手法、および、ツールDBprobe［1－6］を開発している。

このDBprobeは、これまでユーザベンチマークにおけるマルチベンダDBMSの性能の自動実測を可能にし、また、ルール化した性能向上ノウハウを適用することにより、一部の単純なルールではチューニング案を生成し性能測定することを可能にした。

しかし、これらの機能により索引の付け忘れなどの単純なトラブル防止は可能になったものの、未だにDBMSの専門家のチューニングには遠く及ばない。

そこで本稿では、より複雑なチューニングとして複数のSQL文の実行順序に着目したチェック方式について述べ、特定のモデルに適応した際の具体的な例と効果を示す。

2．DBprobeのSQL実行順序チェック方式

ロック待ちの発生や無駄なSQL文の発行など、トランザクションにおけるSQL文の発行順序構成が性能劣化の原因となる場合がある。

このような場合のチューニングはAPの構造に強く依存するために難しい。そこで、AP内のSQL文の実行順序をチェックし、性能上問題となる可能性のある場合に警告を出すことを考える。以下にチェックルールの例を示す。

RULE1:「select結果により同一表に対してupdate, insertを行っている場合、selectの削除ができないか」

RULE2:「最初のSQL文がinsert,deleteの場合、それらをトランザクション後半に移動できないか」

RULE3:「資源のアクセス順を統一できないか」

一般にAP内のSQL文の実行順序を取得するには、以下の方法が考えられる。

(a) 実際にAPを動作させ、実行ログを取得。
(b) APソースプログラムを解析し、実行順序を取得。

しかし、チューニング環境で実際のAPを動かすことが困難だったり、分岐方向が予測できず実行順序の再現ができない、といった問題がある。

そこで、我々はDBprobeの自動実測の特性を利用し、以下のような方法でSQL文の実行順序を自動で取得する。

STEP1 実APからSQL実行順序を疑似する偽APを作成。
STEP2 偽APを実行し、実行順序のログを取得。
STEP3 ログに対してチェックルールによる診断と警告。

DBprobe-check:SQL sequence checker
Yoshihisa NATSUME, Naoya KANOU, Hiroki AKAMA
NTT Information and Communication Systems Labs.

図１に示すAPがRULE１に適合される例をもとに、本方式のアルゴリズムの概要を以下に示す。

【ＳＴＥＰ１】

チューニング対象AP（図１）に以下の操作を行うことで、SQL文実行順序取得のための偽AP（図２）を作成する。

(1-1) 各SQL文に識別子を付与し、ログ取得関数に置換。
(1-2) 各SQL文と識別子の対応をSQL管理ファイルへ保管。
(1-3) SQL文および制御構造（if,while文等）以外の削除。
(1-4) SQL文に関係のない制御構造の削除。
(1-5) 制御構造を仮想制御構造に変換。

```
...
EXEC SQL select count(*) into :H2 from T1 where C1=:H1;
/* データがなければ挿入、あれば更新 */
if ( H2 == 0 ) {
  EXEC SQL insert into T1 values( '1','20','denmwa' );
} else {
  EXEC SQL update T1 set C2=C2+:H3 where C1=:H1;
}
EXEC SQL commit work;
...
```

図１ チューニング対象APの例

一般にソースリストのみから繰り返しの回数や分岐方向を決定するのは困難である。そこで、本手法では制御構造を、ある回数だけ実際に繰り返す、または、実行する毎に異なる方向に分岐するような仮想制御構造に変換する。ただし、この手法は網羅を行わないため完全ではない。以下に、仮想制御構造への変換方法を示す。

・if　　：実行毎に異なる方向へ分岐させる。最低２回の実行で網羅できるので再現回数２とする。
・switch：ifと同様にcase,default数を再現回数とする。
・while ：内側のifとswitchの再現回数の総和だけ繰り返す処理パターンと、全く実行しないパターンを作成する。
・for　　：whileと同様。
・exit　：偽APの先頭へ戻り、実行を継続する。

```
...
{ static fg=0; int i1; int i2; i2=(fg==0)?(fg=1,0):(3);
for( i1=1; i1<=i2; i1++ ) {
  puts( "SQL-1" );
  { static fg=0, stop_then=0, stop_else=0;
  fg=(stop_then>0&&stop_else>0)?(exit()):((stop_then>0)?(0):
       ((stop_else>0)?(1):((fg==1)?(0):(1))));
  if ( fg ) {
    stop_then=1;
    puts( "SQL-2" );
    stop_then=0;
  } else {
    stop_else=1;
    puts( "SQL-3" );
    stop_else=0;
  } }
  puts( "SQL-4" );
} }
...
```

図２ 偽AP

【ＳＴＥＰ２】

実行順序再現のための偽APを実行し、実行順序のログをファイルに取得する（図３）。

```
・・・
プログラム終了
SQL-1
SQL-2
SQL-4
SQL-1
SQL-3
SQL-4
プログラム終了
SQL-1
・・・
```

図３ 実行順序のログ

【ＳＴＥＰ３】

STEP2で取得したSQL文の実行順序ログ、および、(1-2)で作成したSQL管理ファイルを用い、個々のSQLチェックルールを以下のように適合させる。その結果を図４に示す。

(3-1) 以下の全条件を満足する場合、RULE1に適用可能だと判断する。

- ログ中に同一のselectが存在する。
- ログ上でselectの直後のSQL文は、insertの場合とupdateの場合がある。
- select, insert, updateの対象表は同一である。
- select, updateのwhere条件が同一である。

(3-2) RULE1のメッセージを表示する。

「select結果により同一表に対してupdate, insertを行っている場合、selectの削除できないか」

```
・・・
EXEC SQL update T1 set C2=C2+:H3 where C1=:H1;
if ( sqlcode.sqlca == 100 ) {
  EXEC SQL insert into T1 values( '1','20','dennwa' );
}
EXEC SQL commit work;
・・・
```

図４ チューニング結果

３．実験と評価

本方式が、実行順序を考慮しない（単体の）SQL文チェック方式のみの場合に比べ、どの程度、応答時間、スループットについて向上するのか実験を行った。

【実験の方法】

対象Ａ：４つのAPからなる受発注業務型ベンチマーク。
対象Ｂ：対象Ａに単体のSQL文チェックおよびチューニングを適用したもの。
対象Ｃ：対象ＢにSQL文の実行順序を考慮したチェックおよびチューニングを適用したもの。

実験の結果を図５に示す。ただし、平均応答時間は、対象Ａを100%とした相対グラフとし、平均スループットは対象Ｃを100%とした相対グラフとした。

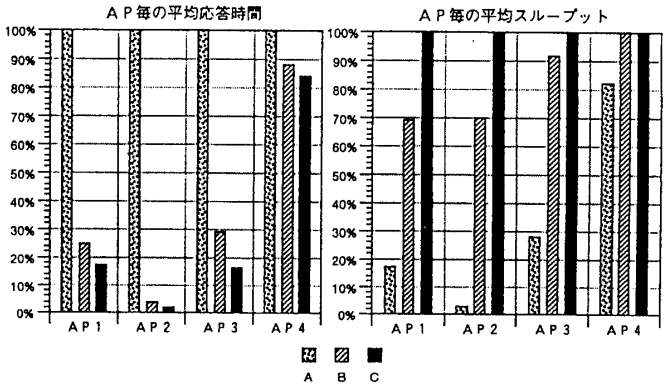

図５ AP毎の平均応答時間とスループット

【評価】

実験の結果では本方式（対象Ｃ）は、単体のSQL文チェック（対象Ｂ）のみによる方式に比べても、応答時間で平均約２９%、スループットで平均約２３%ほど性能が向上している。

このことは、従来の単体のSQL文チューニング後に、まだ改善の余地が充分残されていることを示しており、本稿で提案した方式を加えれば、より効果的なチューニングが可能になることを示している。

４．まとめ

本稿では、我々のDB性能評価システムにおける自動的なSQL文の実行順序チェック方式について述べ、その効果を確認した。

現在は、先に示した３つのルールしかサポートしていないが、今後は、そのルール数をさらに増やしていく予定である。また、実システムに対して適用し、その効果を確認していきたい。

参考文献

[1] 八田、赤間、友野、武田、”ＤＢ性能検証システムDBprobe-link/view”、情処４８全国大会 2F-5
[2] 長谷川、赤間、武田、”ＤＢＭＳ物理情報のチューニング支援システムDBprobe-tune”、情処４８全国大会 2F-4
[3] 石垣、”データベースシステムの性能実測評価環境について”、情処SIG-DBS98-7、 1994.5
[4] 長濱、赤間、八田、”市販データベース管理システムの最適利用に向けて”、NTT技術ジャーナル、1994.10
[5] 赤間、石垣、”ダウンサイジングにおけるデータベース利用の課題と対策”、Computer Today、1994.3
[6] 夏目、八田、赤間、長濱、”ＤＢ性能評価における実システムの疑似方式(DBprobe-reverse)”、情処５０全国大会 5G-5